

A HIGH-CAPACITY SEPARABLE REVERSIBLE METHOD FOR HIDING MULTIPLE MESSAGES IN ENCRYPTED IMAGES

M. Hassan Najafi and David J. Lilja

University of Minnesota, Twin Cities, MN 55455, USA

ABSTRACT

This work proposes a high-capacity scheme for separable reversible data hiding in encrypted images. At the sender side, the original uncompressed image is encrypted using an encryption key. One or several data hiders use the MSB of some image pixels to hide additional data. Given the encrypted image containing this additional data, with only one of those data hiding keys, the receiver can extract the corresponding embedded data, although the image content will remain inaccessible. With all of the embedding keys, the receiver can extract all of the embedded data. Finally, with the encryption key, the receiver can decrypt the received data and reconstruct the original image perfectly by exploiting the spatial correlation of natural images. Based on the proposed method a receiver could recover the original image perfectly even when it does not have the data embedding key(s) and the embedding rate is high.

Index Terms— Image encryption, reversible data hiding, image recovery, privacy preserving.

1. INTRODUCTION

Processing encrypted data can be quite useful for many applications, such as hiding information inside an encrypted image. A common application is a buyer-seller watermarking protocol in which the seller of the multimedia product encrypts the original data using a public encryption key and then embeds a unique fingerprint to identify the buyer inside the encrypted data. A more general case could be situations in which the content owner has encrypted an image but wants to embed more than one additional data stream.

Reversible data hiding (RDH) in images is a technique for embedding additional data into images such that the original cover image can be losslessly recovered after the embedded data are extracted. Tian [1] uses the difference between two consecutive image pixels to embed an additional bit. Ni et al [2] shift the bins of an image histogram to conceal the additional data. Celik et al [3] use a lossless compression technique to create extra space for carrying extra data bits. Thodi et al [4] combines the difference expansion and histogram shifting techniques to embed data. Hong et al [5] and Chang et al [6] also focus on using RDH in the spatial domain.

More recent methods of RDH in encrypted images can be classified into two categories – joint methods in which data extraction and image recovery are performed jointly, and separable methods in which image decryption and data extraction can be performed separately. Zhang [7] introduced a joint method that modifies the least significant bits (LSBs) of the encrypted image to embed additional data. An improvement to this approach [8] uses a side match technique while another variation [9] adapts a pseudorandom sequence modulation mechanism both to enhance the ability to extract the correct embedded data and to extract a better reconstructed image. A problem with all of these joint methods is that, when increasing the embedding rate, the probability of correctly retrieving the embedded bits and recovering the original image decreases significantly.

Zhang [10] proposed a separable method to compress the LSBs of some pixels in the encrypted image to free space for additional

data. A receiver with the embedding key can extract the additional data without error. However, for perfect recovery of the original image, the receiver needs to have both the encryption and the embedding keys. Although the method proposed in [10] guarantees an error-free data extraction, it is not suitable for high embedding payloads. Qian et al [11] use a histogram modification and an n -ary data hiding method. Ma et al [12] and Zahng et al [13] introduced two separable methods that reserve room for data hiding before encryption. Although data retrieving and image recovery in both of these methods are error-free and improve the embedding capacity significantly, making space for data embedding is not always possible. Recently, a progressive recovery method [14] is proposed to improve the embedding rate. This method divides the embedding procedure into three rounds to hide additional messages. However, it supports only one data embedder and both embedding and encryption keys are required to perfectly recover the original image. [15–24] are some other recently proposed methods of data hiding in encrypted image.

This paper proposes a high-capacity separable RDH method for hiding $n \geq 1$ additional data streams inside the encrypted image using n embedding keys. In our proposed method, some pixels of the encrypted image are marked as suitable locations for embedding additional data, and then are divided equally among the data hiders. As result, when fewer embedding keys are needed, more data can be embedded for each key. Another contribution of this method is the guarantee to perfectly reconstruct the image at the receiver side even when there is a high embedding payload and the receiver has only the encryption key.

2. PROPOSED METHOD

The proposed method is made of image preprocessing, image encryption, data embedding, and data extraction/image reconstruction phases. At the sender, the input image is first preprocessed to determine the pixels that are not predictable if they are modified in the embedding phase. The owner of the image then encrypts the original image. One or several data hiders use the most significant bits (MSBs) of selected image pixels, specified by the data embedding key(s), to embed their data. At the receiver side, the embedded data corresponding to each embedding key is decrypted and extracted without needing to know the encryption key. A receiver with the encryption key can directly decrypt the encrypted image containing the additional data. However, since the data hiders changed the MSBs of some image pixels, a recovery step is required to recover the original image. By exploiting spatial correlation between neighboring pixels, the MSBs of all modified pixels can be predicted to reconstruct the original image perfectly. Fig. 1 shows the dataflow of the proposed method.

2.1. Image Preprocessing

Before encrypting the image and embedding the additional data, the original image first needs to be preprocessed to determine an embedding frame that encompasses the pixels that can be used to carry the additional data. The embedding frame is defined to be the largest

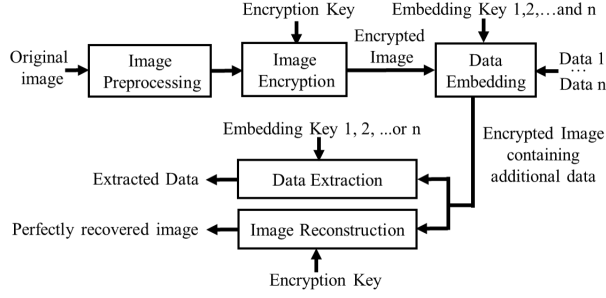


Fig. 1: Data flow of the proposed method

a)	<table> <tr> <td>k</td> <td>D</td> <td>$P1_W, P1_H, \dots, P4_W, P4_H$</td> <td>Private embedding key</td> </tr> </table>	k	D	$P1_W, P1_H, \dots, P4_W, P4_H$	Private embedding key
k	D	$P1_W, P1_H, \dots, P4_W, P4_H$	Private embedding key		
b)	<table> <tr> <td>$P1_W, P1_H, \dots, P4_W, P4_H$</td> <td>Extra bits of encryption key</td> </tr> </table>	$P1_W, P1_H, \dots, P4_W, P4_H$	Extra bits of encryption key		
$P1_W, P1_H, \dots, P4_W, P4_H$	Extra bits of encryption key				

Fig. 2: a) Embedding key (k: key id, D: number of embedding keys, $P1_W \dots P4_H$: embedding frame border pixel locations) b) Encryption key ($P1_W \dots P4_H$: embedding frame border pixel locations)

rectangle in the image that does not include the border pixels and the unpredictable pixels (pixels that are not predictable based on our neighboring prediction method). For a perfect (lossless) recovery at the receiver side, the embedding frame cannot contain any of the unpredictable pixels. However, if it is acceptable to have a few mispredicted pixels at the receiver (lossy recovery), all of the image pixels (excluding the border pixels) can be selected to be within the embedding frame. The trade-off is that for some images, the lossless case has a slightly lower embedding capacity. The output of the preprocessing step is the locations of four pixels that define the border of the embedding frame. These locations need to be sent to any receiver who wants to decrypt the image or extract the additional embedded data. As shown in Fig. 2 parts of the encryption and embedding keys are used for transmitting these locations to the receiver.

2.2. Image Encryption

Assume the original image is an $N1 \times N2$ gray-scale image, each pixel represented by 8 bits. At the sender side, the content owner generates $8 \times N1 \times N2$ pseudo-random bits, each bit corresponding to one bit of the image, using a pseudo-random bit generator which is initialized using the encryption key. The d th bit of the pixel at location (i, j) is encrypted by:

$$e_d(i, j) = b_d(i, j) \oplus r_d(i, j) \quad (1)$$

where \oplus represents the XOR operation, $b_d(i, j)$ and $r_d(i, j)$ are the associated bits in the original image and in the set of generated pseudo random bits, respectively. Without having the encryption key, a receiver cannot generate appropriate pseudorandom bits and so cannot decrypt the received data to obtain the original image.

2.3. Data Embedding

In the data embedding phase, $n \geq 1$ additional data streams can be embedded into the encrypted image using n embedding keys. With each of these embedding keys, the receiver can extract the corresponding data. The process of extracting different embedded data is completely independent, data 1 can be extracted using embedding key 1, followed by data 2 using key 2, or vice versa.

As the first step in the data embedding phase, a collection of locations in the embedding frame needs to be chosen for storing the additional data. Beginning from the first location in the embedding frame and continuing to the last location, we mark every other location in each row and column as a qualified pixel for embedding. For an $F1 \times F2$ frame, the total number of qualified pixels, Q , is

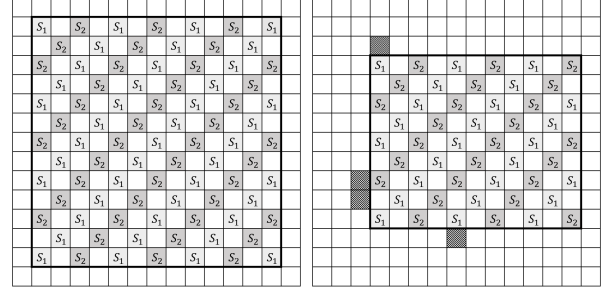


Fig. 3: The embedding frame and qualified locations in (left) a completely predictable 15×15 image ($F1=13, F2=13$, so $Q=85$) (right) a 15×15 image containing four unpredictable pixels ($F1=11, F2=9$, so $Q=50$), when using two keys to embed two different data streams.

$$Q = (\lceil \frac{F1}{2} \rceil \times \lceil \frac{F2}{2} \rceil) + (\lfloor \frac{F1}{2} \rfloor \times \lfloor \frac{F2}{2} \rfloor) \quad (2)$$

Depending on the number of data streams to hide (D) these Q locations are divided into separate groups, with each group used to hide one data stream. The Q qualified locations are assigned to the different data streams by:

$$S_k = \{i \mid (i \bmod D) = k\} \quad i = 1, 2, \dots, Q \quad (3)$$

where S_k is the set of locations assigned to data k . The k and D parameters are both integer values that will be sent to the receiver as part of the data embedding key(s) (Fig. 2.a). With D embedding keys the number of pixels assigned to each data stream is at most Q/D , which then determines the maximum size of each data stream that can be embedded. Fig. 3 shows the embedding frame and the qualified locations for embedding two different data streams in a 15×15 image, one when the image contains no unpredictable pixels, and the other when the image contains four unpredictable pixels.

In the next step, data hiders embed their data into the assigned locations. Assume two data hiders want to embed two data streams with L_1 and L_2 bits, respectively. The first data hider pseudorandomly selects L_1 pixels from S_1 . Let $S(1), S(2), \dots, S(L_1)$ be the L_1 bits of the first data stream and $B(1), B(2), \dots, B(L_1)$ be the L_1 pixels in the encrypted image which are selected to hide and carry this data. Now the first data hider generates L_1 pseudorandom bits using embedding key 1 and performs an XOR operation between the corresponding bits in its data stream and the generated bits. This way an encrypted version of the first data stream is ready to be embedded inside the MSBs of the selected pixels by

$$B_{emb}(d) = B(d) - b * 2^{m-1} + (S(d) \oplus R(d)) * 2^{m-1} \quad (4)$$

where $B_{emb}(d)$ is the modified encrypted pixel associated with the d th bit of the additional data, $R(d)$ is the corresponding pseudorandom bit, b is the MSB of the $B(d)$ pixel, and m is the MSB position. In the same way, the second data hider makes an encrypted version of its data using its embedding key, and then embeds the encrypted version of the data in the assigned permuted locations.

2.4. Data Extraction

In the data extraction phase, a receiver can extract any embedded data by using its corresponding key. The data extractor first locates the embedding frame using the information received about the borders of the frame. It then finds the qualified pixels corresponding to the embedding key using D and k values. Now L_k pixels containing the L_k encrypted bits of data stream k are obtained using embedding key k . Let $B_{emb}(1), B_{emb}(2), \dots, B_{emb}(L_k)$ be the retrieved pixels containing the encrypted version of data stream k . The decrypted version of the L_k embedded bits are computed using

$$S(d) = (\lfloor B_{emb}(d)/2^{(m-1)} \rfloor \bmod 2) \oplus R(d), \quad 1 \leq d \leq L_k \quad (5)$$

Since the process of embedding each additional data stream was independent of embedding other data streams, the process of extracting them is also independent.

2.5. Image Decryption and Recovery

As an improvement to the previously proposed separable RDH methods that needed both the embedding and the encryption keys for perfect recovery of the original image, in our proposed scheme, the encryption key is sufficient to decrypt the encrypted image and reconstruct the original image perfectly. The method proposed in [25] uses the MSBs of some qualified pixels to embed the additional data. When the receiver has only the encryption key, it applies a median filter to the directly decrypted image to recover the original image. As we will see in Section 3, although applying a median filter can improve the quality of the recovered image, some recovered pixels still experience error. In our proposed scheme, we use the averages of four immediate neighboring image pixels to estimate the correct pixel intensities and recover the original image perfectly.

In this phase, the receiver needs to first decrypt all of the received data. So $8 \times N_1 \times N_2$ pseudorandom bits are generated based on the encryption key and then these generated bits are XORed with their corresponding bits in the received encrypted image that contains the additional data. In the decrypted image, the pixels are divided into two categories. The first is the set of qualified locations in the embedding frame which could have been modified in the embedding phase. The second is the set of unmodified locations consisting of the pixels outside of the embedding frame, plus the neighbors of the qualified locations. These pixels are preserved to ensure that they remain the same as in the original image to allow perfect image reconstruction. Starting from the first pixel location in the embedding frame, the receiver adds every other location to the set of qualified locations. Now the receiver estimates the actual value of the pixels in these locations by averaging their four immediate neighbors:

$$D_{est}(i, j) = (D_{dec}(i+1, j) + D_{dec}(i-1, j) + D_{dec}(i, j+1) + D_{dec}(i, j-1)) / 4 \quad (6)$$

where $D_{est}(i, j)$ and $D_{dec}(i, j)$ are the estimated value and the value obtained after direct decryption, respectively, of the pixel at location (i, j) . Since the embedding process embeds the additional bits in the MSB of the image pixels, we also compute the value of the qualified pixels when their MSB is flipped. By having the estimated values of the pixels, their current values after the direct decryption process, and also their values after decryption and flipping their MSBs, the prediction distortion can be computed for both the current and the flipped values using (7) and (8):

$$Distortion_{current}(i, j) = |D_{est}(i, j) - D_{dec}(i, j)| \quad (7)$$

$$Distortion_{flipped}(i, j) = |D_{est}(i, j) - D_{flp}(i, j)| \quad (8)$$

By comparing the computed distortions, the algorithm determines the correct value of the pixel at location (i, j) . If $Distortion_{current}$ was less than $Distortion_{flipped}$, the current value of the pixel after directly decrypting the received data is the original value, otherwise, the flipped value should be used in the recovered image. Note that, in this proposed separable method, the embedded data must be retrieved from the encrypted image but not from the directly decrypted/recovered image. It is evident that in this method there is no need to know the embedding key(s) to recover the original image and the receiver only needs the encryption key and the coordinates of the embedding frame for perfect recovery of the original image.



Fig. 4: (a) Original Lena image; (b) directly decrypted image (PSNR=12.05dB, SSIM=0.05); (c) Recovered image using a median filter (PSNR=23.84dB, SSIM=0.63); (d) Recovered image using the proposed method (PSNR=+∞ dB, SSIM=1.0).

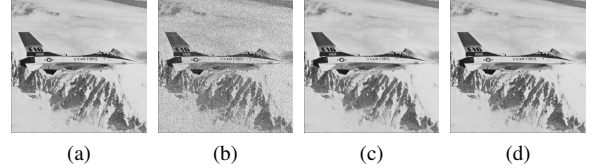


Fig. 5: (a) Original airplane image; (b) Directly decrypted image using encryption key (PSNR=12.02dB, SSIM=0.07); (c) Filtered decrypted image using Wu method [25] (PSNR=32.38dB, SSIM=0.63); (d) Recovered image using our method using only the encryption key (PSNR=+∞ dB, SSIM=1.0).

3. EXPERIMENTAL RESULTS

In order to evaluate the performance of the proposed method we perform some experiments on standard test images. In the reported results, the maximum embedding rate is the maximum number of bits that can be embedded inside the embedding frame of each encrypted image, divided by the total number of pixels in that image. The visual quality of the directly decrypted image and the reconstructed image are evaluated using PSNR (Peak Signal-to-Noise Ratio) and the SSIM (Structural Similarity Index Measure) [26] metrics.

The standard (512×512) Lena image shown in Fig. 4(a) was selected as our main test image for understanding the algorithm. The image preprocessing step showed that there are no unpredictable pixels in the Lena image when we used the four neighboring pixels prediction method. Thus, the embedding frame for the Lena image contains all image pixels excluding the pixels on the borders of the image. At the sender side, all 8 bits of every image pixel are first encrypted using the encryption key and then converted to gray-scale values to generate the encrypted image. Since the qualified pixels in the embedding frame of the encrypted image are selected for carrying different additional bits based on the data embedding key(s), we repeat the experiment 100 times with different embedding keys and different additional data streams. We then report the minimum value of the measured PSNRs and the average value of the calculated SSIMs for the visual quality of the directly decrypted image and the reconstructed image. Using an image size of (512×512) pixels, we are able to embed at most 130050 bits inside the encrypted image. For each iteration of the experiment, we generate two random data streams with $L_1 = L_2 = 65,000$ bits each to be embedded in the pixels using two random embedding keys. This gives a data embedding rate of 0.4959 bits per pixel (bpp).

With an encrypted image containing additional data, a receiver could extract each embedded data stream using its associated embedding key. Direct decryption of the encrypted image containing the embedded data using the encryption key produced Fig. 4(b) with the PSNR=12.05 dB and SSIM=0.05. Since we used the MSB of some image pixels to carry the additional data, the embedding phase introduced salt-and-pepper noise on the directly decrypted image. Wu [25] suggests suppressing this noise using a median filter. Applying the median filter increased the PSNR to 23.84 dB and SSIM

Table 1: Performance analysis and quality evaluation of the proposed method for the two approaches for choosing the embedding frame using ten different 512×512 standard test images.

Test Image	Unpredictable Pixels	Embedding Frame includes all image pixels excluding border pixels					Embedding Frame includes all image pixels excluding border pixels and unpredictable pixels				
		Embedding Frame Size	Embedding Capacity (bits)	Maximum Embedding Rate	Min PSNR	AVG SSIM	Embedding Frame Size	Embedding Capacity (bits)	Maximum Embedding Rate	Min PSNR	AVG SSIM
Lena	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
Airplane	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
Baboon	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
Peppers	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
Camera man	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
house	0	510*510	130050	0.4961	∞	1	510*510	130050	0.4961	∞	1
Pirate	9	510*510	130050	0.4961	50.62	0.999	510*201	51255	0.1955	∞	1
Lake	2	510*510	130050	0.4961	55.40	0.999	510*509	129795	0.4951	∞	1
Barbara	466	510*510	130050	0.4961	33.28	0.992	510*282	71910	0.2743	∞	1
Walk bridge	36	510*510	130050	0.4961	44.37	0.998	510*155	39525	0.1507	∞	1

Table 2: Comparisons to related work.

	Separable	Error in data extraction	Error in image recovery	Image preprocessing	Embedding multiple data streams	Perfect recovery without embedding key	Max embedding rate with lossless recovery (Lenna)
Proposed method	Yes	No	No	Yes (finding frame)	Yes	Yes	0.4959 bpp
Wu's joint method [25]	No	Yes	Yes	No	No	No	0.0625 bpp
Wu's separable method [25]	Yes	No	Yes	No	No	No	0.1563 bpp
Zhang's method [7]	No	Yes	Yes	No	No	No	0.0009 bpp
Hong et al's method [8]	No	Yes	Yes	No	No	No	0.001 bpp
Zhang's method [10]	Yes	No	Yes	No	No	No	0.033 bpp
Ma et al's method [12]	Yes	No	No	Yes (reserving room)	No	No	0.100 bpp
Zhang et al's method [13]	Yes	No	No	Yes (reserving room)	No	No	0.020 bpp
Qian et al's method [14]	Yes	No	No	No	Yes	No	0.0430 bpp
Qian et al's method [15]	Yes	No	No	No	No	No	0.2952 bpp

to 0.63. (Fig. 4(c)). With our method, however, the PSNR and the SSIM of the recovered image increase to ∞ and 1.0, respectively, meaning that the image is recovered perfectly. The reconstructed image using our method is shown in Fig. 4(d).

The second experiment compares the performance of our method and the separable method proposed in [25]. Wu [25] performed an experiment on the airplane image (Fig. 5(a)) using their proposed separable method to embed 40,960 bits (an embedding rate of 0.1563 bpp) in the MSB of some image pixels. Their results show that if the receiver has only the encryption key, the filtered decrypted image will look like the original image with PSNR=32.38 dB (Fig. 5(c)). Note that in their separable method the receiver needs both the encryption and the embedding keys to reconstruct the original image without error. However, as shown in Fig. 5(d), using our separable method to embed the same amount of additional data, the receiver can reconstruct the original image perfectly without even a single error using only the encryption key.

We further analyze our proposed method using eight additional 512×512 gray-scale standard test images. Table 1 shows the number of unpredictable pixels, the embedding frame size and capacity, and the minimum PSNR and average SSIM for the reconstructed images after embedding the maximum possible additional data streams. As can be seen in Table 1, when the embedding frame does not include unpredictable pixels, the output of the image reconstruction step is an image exactly the same as the original input image. For some of these test images, though, the capacity of embedding additional bits has been decreased. However, when the embedding frame contains all image pixels excluding only the border pixels, the values calculated for the SSIM quality metric report nearly perfect recovered images while also providing the highest possible embedding capacity. When using the PSNR quality metric, even having a few mispredicted pixels in the image recovery phase can produce a massive degradation in the quality of the recovered image. Thus, SSIM seems to be a more reasonable quality metric for the cases when a few mispredicted pixels in the image are acceptable.

Considering the capacity of the embedding frame for carrying

additional data in the ten standard test images reported in Table 1, even when the largest rectangle in the image that does not include unpredictable pixels is chosen as the embedding frame, the maximum embedding rates are much higher than the capacity of the current lossless separable methods proposed in the literature. Table 2 compares the key features of our method with recent well-known joint and separable methods of data hiding in encrypted images. Comparing the maximum embedding rate of different RDH methods for lossless recovery of the standard Lena image shows that our proposed method has a higher embedding capacity. In addition, our method is capable of embedding $n \geq 1$ data streams using n embedding keys inside the encrypted image.

4. CONCLUSION

In this paper we proposed a high capacity, separable, RDH method for encrypted images which consists of image preprocessing, image encryption, data embedding, and data-extraction/image reconstruction phases. In the first phase, the image is processed to identify the unpredictable pixels and define an embedding frame. The content owner then encrypts the original image using an encryption key. One or several data hiders permute some prespecified pixels in the embedding frame of the encrypted image using their embedding keys. Each data hider uses the MSB of the assigned pixels in the encrypted image to embed an encrypted version of an additional data stream. In the data embedding phase, the data hider does not necessarily know the original content. At the receiver side, with an encrypted image containing additional data, there will be two different cases. When the receiver has one or some of the data embedding keys, the corresponding embedded data that are encrypted and hidden inside the encrypted image can be extracted. If the receiver has the encryption key, the embedded data cannot be extracted without knowing the embedding keys, but the received data can still be directly decrypted and the original image reconstructed without any errors. The receiver does not need the embedding key(s) to recover the original image perfectly even with high embedding rates.

5. REFERENCES

- [1] Jun Tian, "Reversible data embedding using a difference expansion," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 8, pp. 890–896, Aug 2003.
- [2] Zhicheng Ni, Yun-Qing Shi, N. Ansari, and Wei Su, "Reversible data hiding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 16, no. 3, pp. 354–362, March 2006.
- [3] M.U. Celik, G. Sharma, A.M. Tekalp, and E. Saber, "Lossless generalized-lsb data embedding," *Image Processing, IEEE Transactions on*, vol. 14, no. 2, pp. 253–266, Feb 2005.
- [4] D.M. Thodi and J.J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *Image Processing, IEEE Transactions on*, vol. 16, no. 3, pp. 721–730, March 2007.
- [5] Wien Hong, Tung-Shou Chen, Yu-Ping Chang, and Chih-Wei Shiu, "A high capacity reversible data hiding scheme using orthogonal projection and prediction error modification," *Signal Processing*, vol. 90, no. 11, pp. 2911 – 2922, 2010.
- [6] C.-C. Chang, C.C. Lin, and Y.H. Chen, "Reversible data-embedding scheme using differences between original and predicted pixel values," *Information Security, IET*, vol. 2, no. 2, pp. 35–46, June 2008.
- [7] Xinpeng Zhang, "Reversible data hiding in encrypted image," *Signal Processing Letters, IEEE*, vol. 18, no. 4, pp. 255–258, April 2011.
- [8] Wien Hong, Tung-Shou Chen, and Han-Yan Wu, "An improved reversible data hiding in encrypted images using side match," *Signal Processing Letters, IEEE*, vol. 19, no. 4, pp. 199–202, April 2012.
- [9] Xinpeng Zhang, Chuan Qin, and Guangling Sun, "Reversible data hiding in encrypted images using pseudorandom sequence modulation," in *Digital Forensics and Watermarking*, YunQ. Shi, Hyoung-Joong Kim, and Fernando Prez-Gonzalez, Eds., vol. 7809 of *Lecture Notes in Computer Science*, pp. 358–367. Springer Berlin Heidelberg, 2013.
- [10] Xinpeng Zhang, "Separable reversible data hiding in encrypted image," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 2, pp. 826–832, April 2012.
- [11] Z.Qian, X.Han, and X.Zhang, "Separable reversible data hiding in encrypted images by n-nary histogram modification," *The Third International Conference on Multimedia Technology*, Atlantis Press, p. 869876, 2013.
- [12] K. Ma, Weiming Zhang, Xianfeng Zhao, Nenghai Yu, and Fenghua Li, "Reversible data hiding in encrypted images by reserving room before encryption," *Information Forensics and Security, IEEE Transactions on*, vol. 8, no. 3, pp. 553–562, March 2013.
- [13] Weiming Zhang, Kede Ma, and Nenghai Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, no. 0, pp. 118 – 127, 2014.
- [14] Z. Qian, X. Zhang, and G. Feng, "Reversible data hiding in encrypted images based on progressive recovery," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–1, 2016.
- [15] Z. Qian and X. Zhang, "Reversible data hiding in encrypted images with distributed source encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 636–646, April 2016.
- [16] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, May 2016.
- [17] M. Li, D. Xiao, A. Kulsoom, and Y. Zhang, "Improved reversible data hiding for encrypted images using full embedding strategy," *Electronics Letters*, vol. 51, no. 9, pp. 690–691, 2015.
- [18] W. Zhang, H. Wang, D. Hou, and N. Yu, "Reversible data hiding in encrypted images by reversible image transformation," *IEEE Transactions on Multimedia*, vol. 18, no. 8, pp. 1469–1479, Aug 2016.
- [19] Z. Yin, A. Abel, X. Zhang, and B. Luo, "Reversible data hiding in encrypted image based on block histogram shifting," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 2129–2133.
- [20] X. Zhang, "Reversible data hiding with optimal value transfer," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 316–325, Feb 2013.
- [21] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted jpeg bitstream," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1486–1491, Aug 2014.
- [22] H. Z. Wu, Y. Q. Shi, H. X. Wang, and L. N. Zhou, "Separable reversible data hiding for encrypted palette images with color partitioning and flipping verification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2016.
- [23] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1622–1631, Sept 2016.
- [24] Ming Li and Yang Li, "Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding," *Signal Processing*, vol. 130, pp. 190 – 196, 2017.
- [25] Xiaotian Wu and Wei Sun, "High-capacity reversible data hiding in encrypted images by prediction error," *Signal Processing*, vol. 104, no. 0, pp. 387 – 400, 2014.
- [26] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, April 2004.